WHAT IS CLAIMED IS:

1. An application programming interface (API) for network applications capable of processing packets having source and destination nodes different from the node where the application runs, said API comprising:

first and second data structures associated with a network interface in communication with a network, said first and second data structures being mapped to an operating system and a network application, wherein:

packets to be passed from the operating system to the network application are stored in a buffer and referenced via respective pointers within said first data structure, said first data structure pointers being inserted into said first data structure by said operating system prior to network layer processing, said first data structure pointers being removed by said network application, insertion and removal of said first data structure pointers being asynchronous with respect to each other; and

packets to be processed as received packets by said network layer of said operating system are stored in a buffer and referenced via respective pointers within said second data structure, said second data structure pointers being inserted into said second data structure by said network application, said second data structure pointers being removed by said operating system, insertion and removal of said second data structure pointers being asynchronous with respect to each other.

2. The API of claim 1, further comprising a primitive for creating said first and a second data structures if said first and a second data structures are not available.

3. The API of claim 1, further comprising a primitive for unmapping said first and a second data structures from the network application, said unmapping primitive operating to destroy said first and a second data structures if said data structures are mapped to no other network application.

Sub A12

4. The API of claim 3, wherein:

in the case of said first and a second data structures not being associated with the network interface, packets to be passed between the network and the network interface are processed by the operating system network layer.

5. The API of claim 1, wherein the operating system's network layer implements the Internet Protocol (IP).

Sub A13

6. The API of claim 1, further comprising a primitive for creating said first and a second data structures mapped both to said operating system and said network application, wherein:

non-network packets to be passed from the operating system to the network application are stored in a buffer and referenced via respective pointers within said first data structure, said first data structure pointers being inserted into said first data structure by said operating system, said first data structure pointers being removed by said network application; and

non-network packets to be passed from said network application to said operating system are stored in a buffer and referenced via respective pointers within said second data structure, said second data structure pointers being inserted into said second data structure by said network application, said second data structure pointers being removed by said operating system.

7. The API of claim 6, wherein the operating system maintains in said first data structure at least a predefined number of pointers.

8. The API of claim 6, wherein the API further comprises a primitive to destroy said first and second data structures.

9. The API of claim 1 wherein other network applications are prevented from accessing a buffer from the time said network application removes a pointer to said buffer from said first data structure and inserts a pointer to said buffer into said second data structure.

10. The API of claim 9, wherein each buffer contains an identifier of any network application having exclusive use of the buffer.

11. The API of claim 10, wherein upon termination of a network application, the operating system automatically reclaims buffers that are in the application's exclusive use.

12. The API of claim 1 wherein said first or second data structure is a circular queue.

13. The API of claim 1, further comprising a primitive for placing the network application in a quiescent state until the operating system inserts a pointer into said first data structure.

14. The API of claim 1, further comprising a primitive for placing the network application in a quiescent state until the operating system removes a pointer from said second data structure.

15. The API of claim 1, wherein the node where the network application runs is configured as one of a host, a bridge, a switch and a router.

16. The API of claim 6 wherein other network applications are prevented from accessing a buffer from the time said network application removes a pointer to said buffer from said first data structure and inserts a pointer to said buffer into said second data structure.

17. An application programming interface (API) for network applications, which applications can process packets whose source and destination nodes are nodes different from that where the application runs, said API comprising a primitive for creating a first and a second data structures associated with a specified network interface, if said data structures do not exist, and mapping said data structures both to the operating system and a specified network application, wherein

the specified network interface receives and sends packets from and to a network,

each said packet is stored in a buffer mapped both to the operating system and the specified network application,

the operating system inserts into and the specified network application may remove from said first data structure a pointer to each buffer containing a packet that the operating system's network layer outputs to the specified network interface, before the network interface sends said packets, said insertions and removals being asynchronous with respect to each other, and

the specified network application may insert into and the operating system removes from said second data structure a pointer to each buffer containing a packet that the specified network interface sends to the network, said insertions and removals being asynchronous with respect to each other.

18. The API of claim 17, wherein the API further comprises a primitive for unmapping said data structures from the specified network application and, if said data structures are mapped to no other network application, destroying said data structures.

19. An application programming interface (API) for network applications, which applications can process packets whose source and destination nodes are nodes different from that where the application runs, said API comprising a primitive for creating a first and a second data structures associated with a specified network interface, if said data structures do not exist, and mapping said data structures both to the operating system and a specified network application, wherein

the specified network interface receives and sends packets from and to a network and does not require a coprocessor,

the specified network application requires supervisor privileges,

every packet is stored in a buffer mapped both to the operating system and every network application,

the operating system's network and higher protocol layers do not process any packets that the specified network interface receives or sends,

the operating system inserts into and the specified network application may remove from said first data structure a pointer to each buffer containing a packet that the specified network interface receives from the network, said insertions and removals being asynchronous with respect to each other, and

the specified network application may insert into and the operating system removes from said second data structure a pointer to each buffer containing a packet that the specified network interface sends to the network, said insertions and removals being asynchronous with respect to each other.

20. The API of claim 19, wherein the API further comprises a primitive for unmapping said data structures from the specified network application and, if said data structures are mapped to no other network application, destroying said data structures.